

# PODSTAWY PROGRAMOWANIA W PYTHON

Dzień 7



# AGENDA

## DAY 7

- instalowanie i importowanie modułów
- listy – kopiowanie, list comprehension
- pętle – manipulowanie przebiegiem
- pliki – csv
- os, sys, PIL, smtplib

1.

**import**

ktoś już wykonał za nas pracę

# IMPORT

```
import modul, modul2  
from modul import funkcja1, funkcja2  
from modul import *
```

```
string, datetime, copy, math, decimal,  
random, os, csv, antigravity
```

# FOLDER ROBOCZY

```
import sys  
import os
```

```
print("Ścieżki wyszukiwania Python:", sys.path)  
print("aktualny folder roboczy:", os.getcwd())
```

**Pamiętać** – PyCharm tworzy własne środowisko uruchomieniowe – dodaje do folderów wyszukiwania (sys.path) folder główny projektu, dlatego wskazujemy relatywną do gł. folderu ścieżkę (day6.fun7).

ścieżki wyszukiwania (sys.path) będą inne jeśli plik z pow. kodem uruchomimy:

- a) w PyCharm, oraz
- b) bezpośrednio w konsoli

# FOLDERY WYSZUKIWANIA

Aby rozwiązać problem z importami (jeśli są) możemy:

- umieszczać importowany moduł w tym samym folderze co plik, do którego importujemy
- rozszerzyć sys.path – `sys.path.append(moja_sciezka)`
- moduły wrzucać do folderu, którego ścieżkę dodajemy w zmiennej środowiskowej PYTHONPATH (*na poziomie systemu!*)
- moduł umieścić w folderze bibliotek standardowych Python (folder `lib\site-packages\`) w instalacji Pythona

## 2. PyPI & pip

Menadżer pakietów Python

# PyPI Python Package Index

lista dostępnych pakietów

[pypi.python.org/pypi](https://pypi.python.org/pypi)



# pip

Menadżer pakietów instalowany razem z Python.  
Komendy w wierszu poleceń:

**pip help** – ogólna pomoc

**pip help install** – pomoc dot. polecenia

**pip list** – lista zainstalowanych pakietów

**pip search** – szuka pakietów w repozytorium online

**pip install *pakiet*** – instalowanie modułu

**pip uninstall *pakiet*** - odinstalowanie

**pip list -o** -sprawdzenie nieaktualnych pakietów

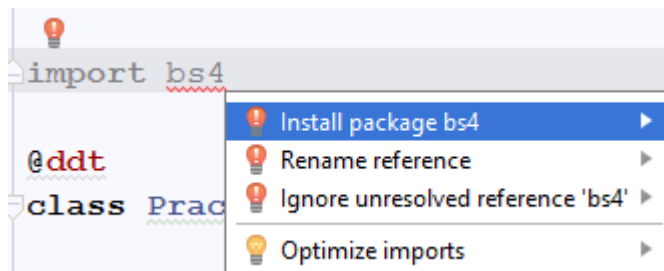
**pip install -U *pakiet*** - update pakietu

**pip freeze > plik.txt** – zapisanie informacji do pliku o pakietach

**pip install -r plik.txt** – zainstaluje wszystkie wymagane pakiety

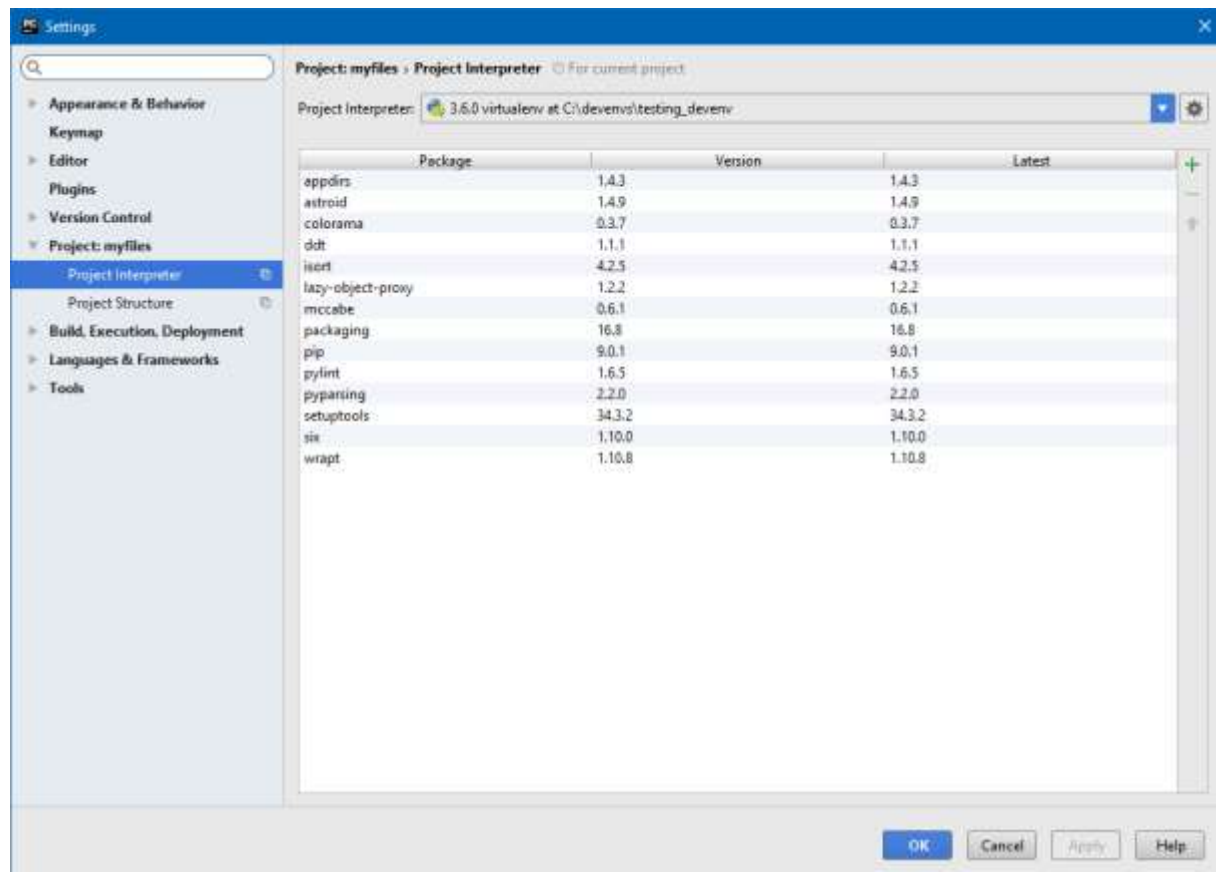
# INSTALOWANIE PAKIETÓW W PYCHARM

- podpowiedzi przy pisaniu kodu – alt + enter



# INSTALOWANIE PAKIETÓW W PYCHARM

file ->  
settings ->  
project ->  
project interpreter



# 3. Tips & Tricks

Kilka przydatnych szczegółów

# 3a. break, continue, else

modyfikacja zachowania pętli

<https://docs.python.org/3/tutorial/controlflow.html#break-and-continue-statements-and-else-clauses-on-loops>

# continue, break

Oba słowa kluczowe używane w pętlach (for, while). Najczęściej w jakiejś instrukcji warunkowej (if-elif-else) wewnątrz pętli., modyfikują działanie pętli:

**continue** – program pomija pozostałe instrukcje w bloku i wraca do sprawdzenia warunku (while) lub do kolejnego elementu (for)

**break** – działanie pętli jest przerywane, program przechodzi do kolejnej instrukcji po całym bloku pętli

# else

## for-else   while-else

Kod wewnątrz tego bloku **else** wykona się jeśli pętla **NIE** została przerwana przez **break**

```
for x in range(10):  
    if x ** 2 > 100:  
        print("Kwadrat większy niż 100")  
        break  
else:  
    print("Brak kwadratu większego niż 100")
```

3.b

# kopiowanie list

czyli początek zabawy z obiektami



# kopiowanie list

**Listy są typami referencyjnymi.**

jeśli przypiszemy listę do innej zmiennej to tak naprawdę przypiszemy adres w pamięci do listy

możemy użyć kopiowania list:

```
nowa_lista = lista.copy()  
nowa_lista = list(stara_lista)  
nowa_lista = stara_lista[:]
```

ale czy to zawsze działa???

# kopiowanie list

**Listy są typami referencyjnymi.**

do głębokiego kopiowania (kopiowanie wszystkiego jako wartość) używamy modułu copy i metody deepcopy()

```
import copy
```

```
nowy = copy.deepcopy(stary)
```

# 3.c

## list comprehension

magiczne tworzenie i uzupełnianie list wartościami

# list comprehension

Problem 1: potrzebuję listę liczbami od 20 do 38

# prosty sposób:

```
kwadraty = []
```

```
for x in range(20, 39):
```

```
    kwadraty.append(x)
```

# magiczny sposób

```
kwadraty = [x for x in range(20, 39)]
```

# list comprehension

Problem 2: potrzebuję listę z kwadratami liczb od 20 do 38

# prosty sposób:

```
kwadraty = []  
for x in range(20, 39):  
    kwadraty.append(x**2)
```

# magiczny sposób

```
kwadraty = [x**2 for x in range(20, 39)]
```

# list comprehension

Problem 3: potrzebuję listę z kwadratami liczb od 20 do 38 jeśli liczba jest podzielna przez 3.

# prosty sposób:

```
kwadraty = []  
for x in range(20, 39):  
    if x % 3 == 0:  
        kwadraty.append(x**2)
```

# magiczny sposób

```
kwadraty = [x**2 for x in range(20, 39) if x % 3 == 0]
```

# 3d. Pliki CSV

# CSV

Pliki CSV – comma separated values – dane oddzielane przecinkami

Imie,Nazwisko,Adres,Telefon

Joanna,Kowalska,Gdansk Przytulna,64 654-65-45

Adam,Nowak,Gdynia Swietojanska,0700325487

Do obsługi plików CSV można użyć biblioteki csv

<https://docs.python.org/3/library/csv.html>



# OS

- moduł os służy do pracy z plikami, ścieżkami, zmiennymi systemowymi

mkdir, chdir, getcwd, unlink, rmdir, listdir, walk

- os.path – działania na ścieżkach  
split, join, abspath

<https://docs.python.org/3/library/os.html>

<https://docs.python.org/3/library/os.path.html>

# shutil

wrapper dla poleceń systemowych, w pewnych sytuacjach ułatwia wykonanie poleceń

copytree, move, rmtree,

<https://docs.python.org/3/library/shutil.html>

# send2trash

przenosi do kosza

```
send2trash.send2trash(plik)
```

# Python praktycznie

You've got mail



# wysyłanie email

moduły

**smtp**lib – Simple Mail Transfer Protocole

**imap**lib – obsługa poczty IMAP

**email.mime.MimeText** – format przesyłania informacji MIME

<https://docs.python.org/3.1/library/email-examples.html>

<https://docs.python.org/3/library/smtplib.html#module-smtplib>

# wysyłanie email

## konfiguracja gmail

Serwer poczty przychodzącej (IMAP):	imap.gmail.com Requires SSL: Yes (Wymaga połączenia SSL: Tak) Port: 993
Serwer poczty wychodzącej (SMTP):	smtp.gmail.com Requires SSL: Yes (Wymaga połączenia SSL: Tak) Requires TLS: Yes (if available) (Wymaga połączenia TLS: Tak (jeśli jest dostępne)) Requires authentication: Yes (Wymaga uwierzytelnienia: Tak) Port na potrzeby połączeń SSL: 465 Port na potrzeby połączeń TLS/STARTTLS: 587
Imię i nazwisko lub Nazwa wyświetlana	Imię i nazwisko
Nazwa konta, Nazwa użytkownika lub Adres e-mail	Twój pełny adres e-mail
Hasło	Twoje hasło do Gmaila

# PSEUDOKOD

importuje biblioteki

mam temat wiadomości i treść wiadomości

tworze mailera

witam się z serwerem smtp – tworzę połączenie

włączam szyfrowanie (bo chce przesłać do serwera login i hasło)

loguję się (podając login i hasło)

wysyłam maila

kończę połączenie z serwerem



# Thanks!!